AMENDMENTS TO THE CLAIMS

The following listing of claims will replace all prior versions and listings of claims in the application:

A2

1. (Original) A method of detecting, recovering from and preventing bogus branch instructions in a microprocessor, the method comprising:

decoding a first macro instruction into at least one micro-op;

writing the at least one micro-op into a decoded micro-op cache;

predicting by branch prediction logic whether the at least one micro-op is a branch;

executing the at least on micro-op;

determining if the at least one executed micro-op is a bogus branch of the first macro instruction; and

continuing processing with a second macro instruction,

wherein if the at least one executed micro-op is determined to be a bogus branch, then the method further comprises:

flagging any other micro-ops which pertain to the at least one executed bogus branch micro-op;

removing the flagged micro-ops for retirement; and

scrubbing a branch prediction logic storage buffer upon which the branch prediction logic is based.

2. (Original) The method according to claim 1, further comprising:

fetching from a main memory the macro instruction.

3. (Original) The method according to claim 1, wherein the at least one micro-op is written into the decoded micro-op cache in an order a branch table buffer predicts that the at least one micro-op should be executed.

4. (Currently Amended) The method according to claim 1, wherein executing the at least one micro-op is in at least one of an [[“]]in-order[[”]] or [[“]]out-of-order[[”]] fashion.

5. (Currently Amended) The method according to claim 1, wherein scrubbing the branch prediction logic storage buffer further comprises at least one of:

deallocating any other micro-ops pertaining to the at least one executed bogus branch micro-op;

deallocating at least one old set which had been overwritten in the decoded micro-op cache by a built instruction[[“]]trace[[”]];

deallocating at least one entry that is related to a branch in at least one old set in the decoded micro-op cache; and

deallocating at least one entry that is related to a branch of at least one old set in the decoded micro-op cache that is downstream from the at least one executed bogus branch micro-op.

6. (Original) The method according to claim 1, further comprising:

determining if the branch has been taken.

7. (Currently Amended) A method of detecting bogus branch instructions in a microprocessor instruction pipeline, the method comprising:

predicting whether a first micro-op is a bogus branch instruction; and

looking [[“]]ahead[[”]] in the instruction pipeline to at least one second micro-op related to the first micro-op,

A2

wherein if the first micro-op is predicted to be a bogus branch, the method further comprises;

attaching a signal flag that indicates a bogus branch to the at least one second micro-op.

8. (Original) The method according to claim 7, further comprising:

decoding at least one macro instruction into the first micro-op and the at least one second micro-op; and

writing the first micro-op and the at least one second micro-op into a decoded micro-op cache.

9. (Original) The method according to claim 7, wherein the prediction of whether the first micro-op is a bogus branch instruction is based on branch prediction logic.

10. (Original) A method of recovering from a bogus branch instruction in a microprocessor instruction pipeline, the method comprising:

determining whether a first micro-op is a bogus branch; and

deallocating from a decoded micro-op cache at least one second micro-op related to the first micro-op.

11. (Original) The method according to claim 10, wherein determining whether the first micro-op is a bogus branch is based on branch prediction logic.

12. (Currently Amended) The method according to claim 10, wherein deallocating from the decoded micro-op cache the at least one second micro-op is accomplished by

checking whether a bogus branch signal [[“]]flag[[”]] has been attached to the at least one second micro-op.

A2

13. (Original) The method according to claim 10, wherein deallocating further comprises at least one of:

removing the specific bogus branch;

removing all branches in a set with the bogus branch;

removing all branches in the decoded micro-op cache; and

clearing the entire decoded micro-op cache.

14. (Original) A method of preventing a bogus branch instruction from being executed in a microprocessor instruction pipeline, the method comprising:

writing at least one micro-op into a decoded micro-op cache;

retiring the at least one micro-op; and

scrubbing a branch prediction logic storage buffer.

15. (Original) The method according to claim 14, wherein retiring the at least one micro-op comprises at least:

determining what the actual result for the retired at least one micro-op was.

16. (Original) The method according to claim 14, wherein scrubbing the branch prediction logic storage buffer comprises at least:

comparing what an actual result of the retired at least one micro-op is to an instruction trace in the branch prediction logic storage buffer.

17. (Currently Amended) The method according to claim 14, wherein scrubbing the branch prediction logic storage buffer further comprises at least one of:

deallocating any other micro-ops pertaining to the at least one retired micro-op;

deallocating at least one old set which had been overwritten in the decoded micro-op cache by a built instruction [[“]]trace[[”]];

deallocating at least one entry that is related to a branch in at least one old set in the decoded micro-op cache; and

deallocating at least one entry that is related to a branch of at least one old set in the decoded micro-op cache that is downstream from the at least one retired micro-op.

18. (Original) The method according to claim 14, wherein scrubbing can be accomplished at the time of at least one of writing or retiring.

19. (Original) An apparatus for detecting, recovering from and preventing bogus branch instructions in a microprocessor, the method comprising:

a decoded micro-op cache into which are written at least one decode micro-op of a macro instruction;

a branch prediction logic storage buffer for predicting whether a branch will be taken upon execution of the at least one decoded micro-op;

an instruction execution unit for executing the at least one micro-op; and

an instruction retirement unit which determines whether the at least one micro-op is of a bogus branch macro instruction,

wherein if the instruction retirement unit determines the at least one micro-op is of a bogus branch macro instruction,

any other micro-ops stored in the decoded micro-op cache pertaining to that bogus branch macro instruction are flagged and removed to the instruction retirement unit for retirement

and the branch prediction logic storage buffer is scrubbed.

20. (Original) The apparatus according to claim 19, further comprising:

A2

a main memory in which the macro instruction is stored; and

an instruction fetch unit for fetching the macro instruction from the main memory.

21. (Original) The apparatus according to claim 19, further comprising:

an instruction decode unit for translating the macro instruction into the at least one decoded micro-op.

22. (Original) The apparatus according to claim 19, further comprising:

a jump execution unit which determines whether a branch was taken upon execution of the at least one decoded micro-op.

23. (Original) The apparatus according to the claim 19, wherein the branch prediction logic storage buffer applies branch prediction logic to predict whether a branch will be taken upon execution of the at least one decoded micro-op.

24. (Currently Amended) The apparatus according to claim 19, wherein if the branch prediction logic storage buffer predicts a branch will be taken upon execution of the at least one decoded micro-op, an instruction [[“]]trace[[”]] is built pertaining to the predicted branch.

25. (Currently Amended) The apparatus according to claim 24, wherein the built instruction [[“]]trace[[”]] is inserted into the decoded micro-op cache such that the micro-ops of the branch macro-instruction are executed.

26. (Currently Amended) The apparatus to claim 19, wherein the branch prediction logic storage buffer is scrubbed by deallocation of at least one of

A2

any other micro-ops pertaining to the bogus branch macro instruction,

any old set which had been overwritten in the decoded micro-op cache by a built instruction [[“]]trace[[”]],

all entries that are related to any branches in the old set, and

all entries that are related to the branches in the old set that are downstream from the retired branch macro instruction.